We claim:

1. A method of generating an intermediate representation of program code, the method comprising the computer implemented steps of:

generating a plurality of register objects representing abstract registers, a single register object representing a respective abstract register; and

generating expression objects each representing a different element of said program code as that element arises in the program code, each expression object being referenced by a register object to which it relates either directly, or indirectly via references from other expression objects.

2. A method according to claim 1, wherein said program code is expressed in terms of an instruction set of a subject processor.

3. A method according to claim 2, wherein said register objects represent abstract registers corresponding to registers of said subject processor.

4. A method according to claim 1, wherein each of said steps are performed sequentially for basic blocks of said program code having only one effective entry point instruction and one effective exit point instruction.

5. A method according to claim 1, wherein at least some said expression objects feed into more than one said register object.

6. A method according to claim 1, wherein said expression objects are not duplicated.

7. A method according to claim 1, wherein a single said expression object is generated for a given element of said program code, and each said expression object is referenced by all said register objects to which it relates.

8. A method according to claim 1, wherein if a said register object or a said expression object becomes redundant or unnecessary it is eliminated.

9. A method according to claim 8, wherein a redundant or unnecessary said register object or said expression object is identified by maintaining an ongoing count of references being made to that object as a network of register and expression objects is constructed.

10. The method according to claim 9, wherein for each expression object a count is maintained of the number of references to that expression object from other expression objects or from register objects, the count associated with a particular expression object being adjusted each time a reference to that expression object is made or removed.

11. A method according to claim 10, wherein an expression object and all references from that expression

object are eliminated when said count for that expression object is zero.

12. The method of claim 1, comprising translating the program code written for execution by a processor of a first type so that the program code may be executed by a processor of a second type, using the generated intermediate representation.

13. The method of claim 12, wherein said translating step is performed dynamically as the program code is run.

14. The method of claim 1, comprising optimising the program code by optimising the generated intermediate representation.

15. The method of claim 14, wherein said optimising step is used to optimise the program code written for execution by a processor of a first type so that the program code may be executed more efficiently by that processor.

16. A method for generating an intermediate representation of program code written for running on a programmable machine, said method comprising:

(i) generating a plurality of register objects for holding variable values to be generated by the program code; and

(ii) generating a plurality of expression objects representing fixed values and/or relationships between

said fixed values and said variable values according to said program code;

said objects being organised into a branched tree-like network having all register objects at the lowest basic root or tree-trunk level of the network with no register object feeding into any other register object.

17. A system for generating an intermediate representation of program code, comprising:

means for generating a plurality of register objects representing abstract registers, a single register object representing a respective abstract register; and

means for generating expression objects each representing a different element of said program code as that element arises in the program code, each expression object being referenced by a register object to which it relates either directly, or indirectly via references from other expression objects.

18. A system for generating an intermediate representation of program code written for running on a programmable machine, the system comprising:

means for generating a plurality of register objects for holding variable values to be generated by the program code; and

means for generating a plurality of expression objects representing fixed values and/or relationships between

said fixed values and said variable values according to said program code;

wherein said objects are organised into a branched tree-like network having all register objects at the lowest basic root or tree-trunk level of the network with no register object feeding into any other register object.